Planning and Scheduling of Steel Plates Production. Part II: Scheduling of Continuous Casting

Junichi Mori, Vladimir Mahalec*

Department of Chemical Engineering, McMaster University, 1280 Main St. West, Hamilton, ON, L8S 4L8, Canada

* Corresponding author. Tel.: +1 905 525 9140 ext. 26386. E-mail address: mahalec@mcmaster.ca

Keywords: Steel plates production; Planning and scheduling; Algorithm for sequence optimization; Parallel simulated annealing algorithm; Shuffled frog-leaping algorithm.

Abstract

Production planning and scheduling in the steel industry are challenging problems due to large number of products being produced. This work deals with scheduling of the continuous casting of steelmaking, i.e. determination of charge sequence in each casting machine. Since a full-space model Mixed-Integer Linear Programming (MILP) approach is computationally intractable for long term continuous casting scheduling due to a huge number of binary variables, we propose a two-level optimization method. At the top level, we solve the planning problem that determines the number of pots of each grade for every planning period by solving the relaxed mixed integer linear problem. At the lower level, the scheduling problem is solved by an algorithm which combines ideas from parallel simulated annealing and shuffled frog leaping. Real-world steel-plate production data are utilized to examine the effectiveness of the proposed two-level optimization algorithm.

1. Introduction

Steel plates are high variety low-volume products manufactured on order due to a huge number of different applications. These characteristics of the process make it difficult to obtain optimal production schedules which will increase profit, reduce production costs and material consumption, satisfy the customer specifications and meet the shipping due dates. Simplified process flow of the steel plates production is shown in Fig. 1. A *charge* is the basic unit of steel making production and it consists of the same steel grade. In the continuous casting machine, the casting from charge to charge is carried out to transform molten steel into solid. The first solid form of the steel produced in these facilities is cut up into smaller solid form called *slab*. These slabs are rolled and cut into plates of the correct thickness and properties. Finally, these plates are handled in the finishing and inspection lines for satisfying the customer demands, and then they are sent to the distribution warehouse.

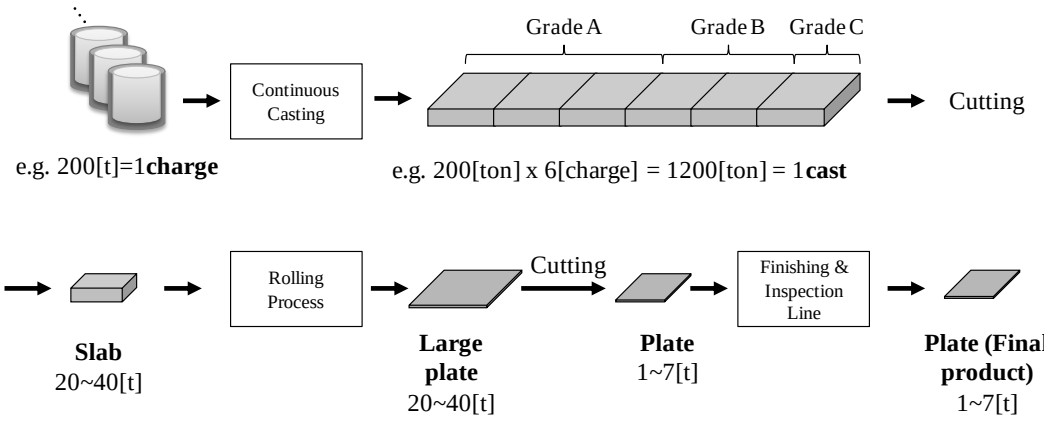

Fig. 1 Process flow of steel-plate production system

There are many scheduling and planning tasks to be completed in the successful production in steel mills. Among them, scheduling of steelmaking continuous casting (SCC) of steel slabs

followed by rolling to steel plate is one of the most important tasks. The main reason is that the SCC process causes most parts of material loss and that the downstream production schedules such as production flows in the steel mills strongly depends on the schedules of the SCC process.

Schedules for steelmaking continuous casting (SCC) production should meet the following criteria:

(1) Each plate is completed before its due date,

(2) Amounts of leftover and contaminated steel which are total loss are minimized,

(3) Continuous casting utilization is maximized (i.e. the residual capacity at the continuous casting stage is minimized),

(4) Production constraints and inventory capacities at the downstream production stage are satisfied.

At the same time, scheduling of SCC must take into account the performance of the downstream processes. Since it is imperative to take into account the customer due date and production capacity of downstream processes such as finishing and inspection lines, it is necessary to predict as accurately as possible the production loads and production times and then to incorporate these prediction models into the mathematical models for optimization. In order to obtain the accurate knowledge of production loads and production times, in the previous work we developed Bayesian network based prediction models for estimation of probability distributions of production loads and process times from the observed variables such as customer demands and operating conditions (Mori & Mahalec, 2015). In this paper, we make use of the Bayesian network model in optimization of SCC scheduling for steel-plate production.

Since SCC process plays an important role in steelmaking industries, many optimization models and approaches have been proposed. In general, SCC production scheduling problems

should determine charge sequencing in each casting machine (first level scheduling) and timing of the charges on continuous casting machines (second level scheduling).

Second level scheduling has been a subject of work by several research groups. Atighehchian et al. (2009)scheduled the second level by an algorithm (HANO) which is a combination of ant colony optimization (ACO) and non-linear optimization. Their objective was to reach production continuity, to increase productivity, and to cut down the total production costs. Just-In-Time (JIT) nonlinear models which consider punctual delivery and production operation continuity have also been proposed to eliminate the machine conflicts given cast sequences (Tang et al., 2000). Discrete-time mixed integer linear programming (MILP) formulation for scheduling of the SCC process has also been developed (Tang et al., 2002). Above methods find optimal timing of the charges on continuous casting machines while the charge sequence in each cast is assumed to be known a prior and determined from first level scheduling.

Although the second level scheduling problems have been well studied, few studies have dealt with the first level scheduling because it is intractable to solve the first level scheduling of the SCC process for a long time horizon (e.g. 1week). Therefore, the first level scheduling are usually determined empirically by skillful persons or computed based on the rule-based approaches.

One approach to solving the first level scheduling is a, decomposition approaches. For instance, Harjunkoski and Grossmann (2001) tackled the first level scheduling of a steel production using mathematical programming methods based on a decomposition scheme that generates smaller sub-problems. However, the presented method does not consider the individual charge which is a unit of production consisting only one grade of steel. Therefore, the

presented method may not be applicable for scheduling of general SCC processes. In addition, all of the existing approaches including both the first and second level scheduling methods do not take into account the production capacities of the downstream processes at all. As we discussed in the previous work (Mori & Mahalec, 2015), the production capacities of downstream processes such as finishing and inspection lines should be taken into account while scheduling, because the downstream processes often create a bottleneck for the entire steel-plate production.

In order to overcome these limitations, we propose a novel approach to the first level scheduling of the SCC processes for steel plate production. Most likely processing times and production capacities for the steel plate manufacturing are first estimated from the Bayesian network of the process. These are then used in the scheduling of continuous casting. The remaining obstacle to solving this scheduling problem is a large number of binary variables that are needed to represent exactly the process. Therefore, a full space model mixed-integer linear programming (MILP) approaches require excessive computational times and often fail even to find a feasible solution within practical execution times.

In order to overcome this limitation, we propose a decomposition approach where the full-space model is divided into two levels: (i) Top level which determines the number of pots per grade for each day by solving the relaxed mixed-integer linear model that does not take into consideration sequence penalties (e.g. contaminated steel) and (ii) Lower level which optimizes the cast sequencing while taking into consideration the sequence penalties in the continuous casting machine by using meta-heuristic methods. Since meta-heuristic approaches are employed in this work, our approach is not guaranteed to yield the global solutions, but they

assure computation of good schedules even for very large problem within reasonable execution times.

Many kinds of meta-heuristic approaches have been proposed. The approaches fall into two categories, which are: single solution and population-based approaches. The single solution approaches such as simulated annealing (Aarts & Korst, 1989), tabu search (Glover, 1986) and iterated local search (Lourenco et al., 2002) focus on modifying and improving a single candidate solution. Meanwhile, the population-based approaches such as genetic algorithm (GA) (Lawrence, 1991), particle swarm optimization (PSO) (Kennedy & Eberthart, 1995), ant colony optimization (ACO) (Marco & Birattari, 2010) and shuffled frog-leaping algorithm (Eusuff and et al., 2006) maintain and improve multiple-candidate solutions and often generate new solutions by population characteristics. In this work we introduce a new parallel meta-heuristic algorithm to schedule continuous casting.

The organization of this article is as follows. Section 2 provides the problem statement while section 3 describes a Bayesian network based mixed-integer linear programming (MILP) model for the scheduling of SCC. In section 4 we introduce the two-level scheduling approach that divides the original full space model into two level scheduling models. In order to solve the lower level scheduling problem efficiently, we propose the novel parallel SA that can avoid local optima as much as possible by communicating states more effectively than the traditional parallel SA by using concepts from shuffled frog algorithm. The presented methods are applied to the real-world steel plate production data in section 5. Finally, the conclusions are presented in section 6.

2. Problem Statement

We assume that the steel plate manufacturing facility has received orders for the next 5 to 10 days or more. Since the number of different steel plate SKU can run into hundreds or thousands, we will group them into grades, based on the qualities of the steel from which they are made of. The task is to determine the amount of products in each charge of each continuous caster over period of 5 to 10 days, as well as the sequence of charges. This is called *Long Term Scheduling*. Once the amount of products in each charge is determined, each plate is assigned to one slab such that the leftover can be minimized, which is called *slab design problem* (Schaus et al., 2010; Dash et al. 2007). Finally, the second level scheduling computes the timing of the charges on continuous casting machines from the results obtained in the first level scheduling. Our paper focuses on the first level scheduling. The diagram of the entire set of scheduling tasks for steel-plate production is shown in Fig. 2.

Our targeted scheduling problem of SCC for steel-plate production is stated as follows:

Given:

1. A scheduling period (e.g. 1 week).
2. The number of charges per cast (e.g. 6 charges per cast).
3. The number of casts per day (e.g. 3 casts per day).
4. The maximum processing capacity of each finishing and inspection process.
5. The size of charges (e.g. 200[ton]).
6. The contamination cost when switching grades between consecutive charges.
7. A set of delivery orders for each grade along the scheduling horizon (e.g. demand profile).

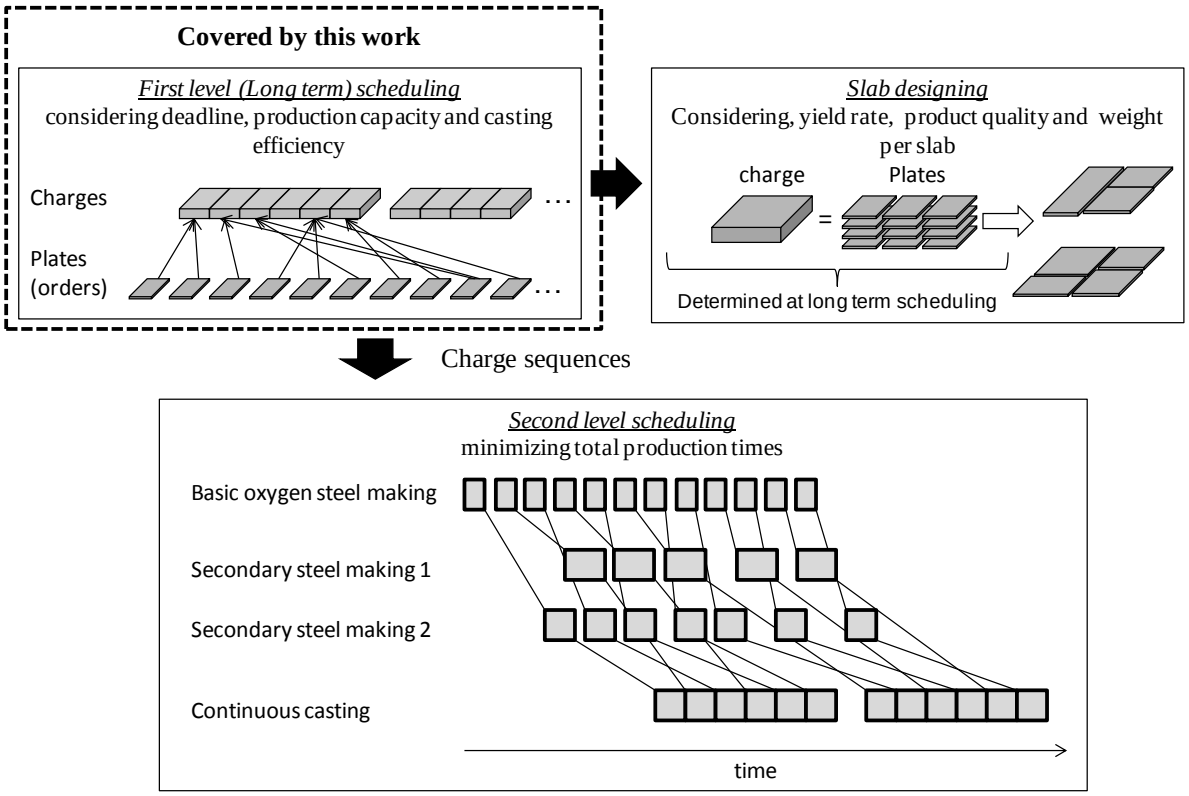

Fig. 2 Illustrative diagram of SCC scheduling for steel-plate production

Determine:

1. The amount of products on each charge of each continuous caster.
2. Sequence of charges
3. The amount of products that each finishing and inspection line should process each day.
4. The inventory profile of products at a distribution warehouse.

While Minimizing:

1. The cost of producing contaminated steel caused by the switching grades between consecutive charges.
2. The cost of leftover in each charge.

3. The amount of inventory at a distribution warehouse.

Subject to:

1. Only one steel grade can be used in a single charge. Once it begins charging, the same grade steel must be used and its amount should always be equal to the size of a charge.
2. Finishing and inspection processes can handle at most the same amount of steel as its processing capacity.

Assuming:

1. There is only one casting machine for steel plates and it cannot be used for other kinds of products such as steel sheet.
2. Processing order of the finishing and inspection lines is fixed.
3. Variables associated with both customer demands and operating conditions are known.
4. Variables associated with both production loads and process time are unknown.
5. Production loads and process times follow probability distributions are known (they are predicted by the Bayesian network model from variables associated with both customer demands and operating conditions.

3. Bayesian Network Based Scheduling Model

Bayesian network model

Our mathematical model focuses on the allocation of grades to each charge such that each plate is completed by its due date, the total leftover in charges and contaminated steel in the continuous casting machine are minimized and that the production and the inventory at the

downstream production lines are less than their capacities. In order to take into consideration the customer due dates and production capacities of the finishing and inspection processes, we need to predict the total production time of each plate. This time will be used for determining its ideal production starting time and the production loads of each plate for each production unit.

Since prediction of the probability distributions of production times for all plates are necessary, we utilize the Bayesian network based prediction model which has been developed in the previous work (Mori & Mahalec, 2015). Based on the Bayesian network model, the total demand $D_{i,k}$ of grade i on the ideal production starting day k and the expectation of production load of grade i at process unit l, elapsed day m can be computed.

Since we assume that the processing order of the finishing line is fixed, without loss of generality the process ordering can be described as unit 1, unit 2, … , unit L with L being the number of the finishing and inspection processes. The parameters of probability distribution $DistA(e; p, l) = N(t; \mu^A_{p,l}, \sigma^A_{p,l})$ of elapsed time e between the times when plate p is started manufacturing and when it is arrived unit l can be computed as follows:

$$\mu^A_{p,l} = \mu^A_{p,l-1} + \mu_{p,l} \tag{1}$$

$$\sigma^A_{p,l} = \sqrt{{\sigma^A_{p,l-1}}^2 + {\sigma_{p,l}}^2} \tag{2}$$

with $\mu_{p,l}$ and $\sigma_{p,l}$ being means and standard deviations of production time of plate p at unit l. It should be noted that both of them can be estimated from variables related to the customer demands and operating conditions using the constructed Bayesian network model. The expectation of production load of plate p and unit l and elapsed day m is described below:

$$Expectation(m; p, l) = \int_{m}^{m+1} \Pr ob(p, l) DistA(e; p, l) de \quad (3)$$

where *Prob*(*p*,*l*) is a probability distribution of production load of plate *p* at unit *l*. The probability distribution $DistT(m, p)$ of total production time from starting manufacturing to arriving at the distribution warehouse can be computed as $DistA(m; p, L+1)$.

The proposed predicting framework for elapsed time from start manufacturing is illustrated in Fig. 3. In this example, we assume that there are three units whose process ordering is Unit 1, Unit 2, Unit 3, and a distribution warehouse. First we need to derive the Bayesian network model that can predict probability distributions of the production time. We learn both the network structure and the model parameters from the historical process data. The detailed algorithms are described in the previous work (Mori & Mahalec, 2015). With the constructed Bayesian network model, we infere the probability distirbution of the production times of all units for each plate by using the information of customer demands and operating conditions. Then, we compute the elapsed time for each unit between the times when the plate manufacturing is started and when it arrives at the corresponding unit. Finally, the total production time $DistT(m, p)$,which is same as the time from start manufacturing to arriving at the warehouse, can be computed from the probability distribution of both the elapsed time so far and the production time of Unit 3.

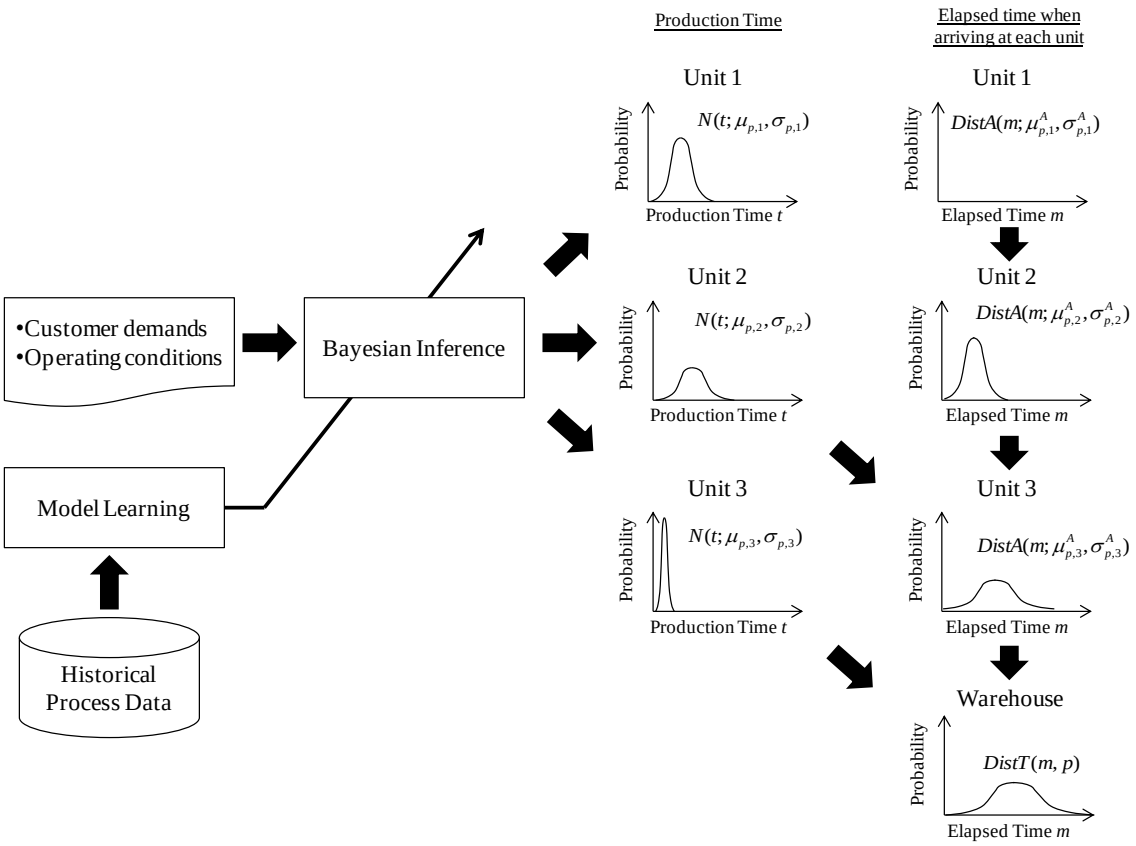

Fig. 3 Illustrative diagram of predicting the elapsed time when arriving at each unit

Given customer due date tc_p which is given for each plate p as demand profiles, an ideal production starting day at the confidence level of α can be obtained below:

$$ts_p = tc_p - r_p \tag{4}$$

where r_p is the estimated production time satisfying below:

$$\alpha = DistT(m \le r_p; p)\,. \tag{5}$$

Finally, the total demand $D_{i,k}$ of grade i on the ideal production starting day k and the expectation of production load of grade i at process unit l, elapsed day m can be written as follows:

$$D_{i,k} = \sum_{p \in grade(i), ts_p = k} weight_p \tag{6}$$

$$Load_{l,i,m} = \frac{1}{N_i} \sum_{p \in grade(i)} Expectation(m; p, l) weight_p \tag{7}$$

where $weight_p$ represents the weight of plate p, grade(i) is the set of plates whose grade is i and N_i is the total weight of all plates belonging to the steel grade i computed as follows:

$$N_i = \sum_{p \in grade(i)} weight_p . \tag{8}$$

Full Space Model MILP

The allocation of grades on each charge can be formulated as follows. The objective function Eq. (9) minimizes the total leftover in charges (f_1^+), the amount of molten steel that violates cast capacity (f_1^-), the amount of contaminated steel (f_2), the amount of steel-plate inventory (f_3), the amount of steel plates which violate maximum inventory capacity (f_3^+), the amount of steel plates which violate the minimum inventory capacity (f_3^-), and the amount of steel plates which violate the capacities of production unit (f_4^-). The values of the coefficient ($W_1^+, W_1^-, W_2, W_3, W_3^+, W_3^-, W_4^-$) depend on the problem, but generally W_1^-, W_4^- are set to be much

greater than the other coefficients to ensure that the final solution satisfies the capacity constraints (if a physical feasible solution exists).

$$\min \quad W_1^+ f_1^+ + W_1^- f_1^- + W_2 f_2 + W_3 f_3 + W_3^+ f_3^+ + W_3^- f_3^- + W_4^- f_4^- \tag{9}$$

All constraints considered in the full space MILP are presented below.

The indices, parameters and variables used in this work are shown in the Nomenclature section. In order to take into account the capacity of charge *P* and penalty of contaminated steel, two binary variables $Y_{i,u,v,k}$ and $Z_{i,i',u,v,k}$ are required. $Y_{i,u,v,k}$ is equal to one if grade *i* is assigned to charge *v* of cast *u* on time period *k*, and otherwise 0. Defining $X_{i,u,v,k}$ as the variables for the amount of steel of grade *i* assigned to charge *v* of cast *i* on time period *k* and slack variables $S1^+_{i,u,v,k}$ and $S1^-_{i,u,v,k}$, we can derive Eqs. (10)(11)(12) which represent the relation among these variables.

$$X_{i,u,v,k} + S1^+_{i,u,v,k} - S1^-_{i,u,v,k} = PY_{i,u,v,k} \quad \forall i,u,v,k \tag{10}$$

$$Z_{i,i',u,v,k} \geq Y_{i,u,v,k} + Y_{i',u,v-1,k} - 1 \quad \forall i,u,k,v \geq 2 \tag{11}$$

$$\sum_i Y_{i,u,v,k} = 1 \quad \forall u,k,v \tag{12}$$

$S1^+_{i,u,v,k}$ is equal to the amount of the leftover in charges while $S1^-_{i,u,v,k}$ becomes positive if the amount of steel products of the corresponding charge is greater than the size of charge, which

means that there are no physical infeasible solutions. Then, the objective functions f_1^+, f_1^- and f_2 are obtained below:

$$f_1^+ = \sum_i \sum_u \sum_v \sum_k S1_{i,u,v,k}^+ \tag{13}$$

$$f_1^- = \sum_i \sum_u \sum_v \sum_k S1_{i,u,v,k}^- \tag{14}$$

$$f_2 = \sum_i \sum_{i'} \sum_u \sum_v \sum_k C_{i,i'} Z_{i,i',u,v,k} \tag{15}$$

where $C_{i,i'}$ is the contamination cost when grade i is assigned after grade i' and its diagonal elements are always zero. Although the variables $Z_{i,i',u,v,k}$ are binary variables, they can be treated as continuous variables because the left-hand side of Eq. (11) must be either -1, 0 or 1 and $Z_{i,i',u,v,k}$ is a positive variable.

The following constraints define the inventory balance for each time period. Eq. (16) means that the amount of input inventory $M_{i,k}^{in}$ is equal to the sum of amount of production $X_{i,u,v,k}$ in the same time period. Eq. (17) simply states that the total demand $D_{i,k}$ of grade i and ideal production time k is equal to the output inventory $M_{i,k}^{out}$. Eq. (18) defines that the closing inventory $MI_{i,k}^{close}$ should be equal to the opening inventory $M_{i,k+1}^{open}$ on the next time period. Eq. (19) considers that the sum of input inventory $M_{i,k}^{in}$ and opening inventory $M_{i,k}^{open}$ should be equal to the sum of $M_{i,k}^{out}$ and $MI_{i,k}^{close}$.

$$M_{i,k}^{in} - \sum_{u}\sum_{v} X_{i,u,v,k} = 0 \qquad \forall i,k \tag{16}$$

$$M_{i,k}^{out} - D_{i,k} = 0 \qquad \forall i,k \tag{17}$$

$$MI_{i,k+1}^{open} - MI_{i,k}^{close} = 0 \quad \forall i,k \tag{18}$$

$$M_{i,k}^{in} + MI_{i,k}^{open} - MI_{i,k}^{out} - MI_{i,k}^{close} = 0 \quad \forall i,k \tag{19}$$

In order to avoid infeasible solutions, we define slack variables $S3_{i,k}^{+}$ and $S3_{i,k}^{-}$ which enables us to state the maximum and minimum inventory constraints as follows:

$$MI_{\max} - MI_{i,k}^{close} + S3_{i,k}^{+} \geq 0 \quad \forall i,k \tag{20}$$

$$MI_{\min} - MI_{i,k}^{close} - S3_{i,k}^{-} \leq 0 \ . \tag{21}$$

$S3_{i,k}^{+}$ is equal to the amount of the inventory of grade *i* and time period *k*, while $S3_{i,k}^{-}$ is zero if there is no delivery delay, otherwise positive. The objective functions f_3, f_3^{+} and f_3^{-} are obtained below:

$$f_3 = \sum_{i}\sum_{k} MI_{i,k}^{close} \tag{22}$$

$$f_3^+ = \sum_i \sum_k S3_{i,k}^+ \tag{23}$$

$$f_3^- = \sum_i \sum_k S3_{i,k}^- \tag{24}$$

The last two constraints consider the production capacity of finishing and inspection processes. Eq. (25) states that sum of the product of the expectation of production load $Load_{l,i,m}$ defined in Eq. (7) and the amount of production $X_{i,u,v,k'}$ over grade *i*, cast *u*, charge *v* and time period *k'* satisfying $k'+m=k$ is equal to the production load $Q_{l,k}$ of process *l* on the time period *k*. Eq. (26) considers that the production load $Q_{l,k}$ should be less than the production capacity $Load_l^{\max}$.

$$\sum_i \sum_u \sum_v \sum_{k',m:k'+m-1=k} X_{i,u,v,k} Load_{l,i,m} - Q_{l,k} = 0 \qquad \forall l,k \tag{25}$$

$$Q_{l,k} - S4_{l,k}^- \le Load_l^{\max} \qquad \forall l,k \tag{26}$$

where $S4_{l,k}^-$ is a slack variable and becomes positive value if the total amount of products that should be handed in the unit *l* and time period *k* is greater than the corresponding production capacity. The objective function f_4^- is obtained below:

$$f_4^- = \sum_l \sum_k S4^- \tag{27}$$

The full space model given above includes a huge number of integer variables and thus it is tractable only for a small number of grades or a brief planning period, as shown in Section 5.

4. Long Term Scheduling of Steel Plates Production by Two Level Algorithm

Due to a large number of binary variables using the full space MILP approach described in section requires an expensive computational effort, even for a scheduling problem with only one day of period. In order to be able to compute good scheduling solutions within acceptable execution times, we decompose the problem in two levels (see Fig. 4):

- Top level determines the number of pots per grade for each day. The main purpose of this level is to assign products to charges so that the customer due dates, capacity constraints and inventory constraints are met. At this level we generate a relaxed integer formulation that does not take into account the sequencing penalty due to contaminated steel between different grades. Therefore, we call the top level *Production Planing.* Although there are still integer variables in the relaxed formulation, this planning problem is much easier to solve than the original full space straightforward formulation.
- At the lower level we optimize the sequence of these charges while taking into account the sequence penalty. We call the lower level *Production Scheduling*.

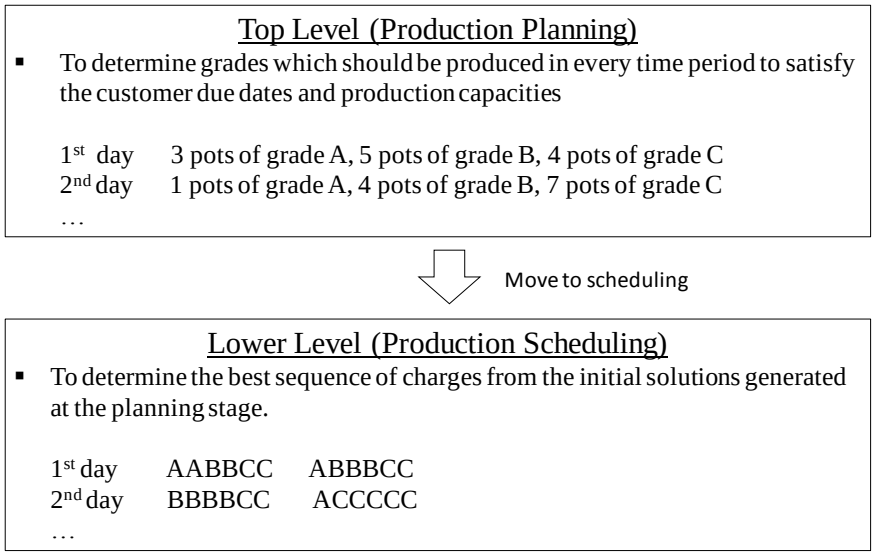

Fig. 4. Two level algorithm for steel plate production scheduling

4.1 Planning Model

Since the sequence penalty is not considered in the relaxed mixed integer linear model that will be described in this section, we introduce a new continuous variable $Xr_{i,k}$ and an integer variable $Ur_{i,k}$ instead of $X_{i,u,v,k}$ and $Y_{i,u,v,k}$. $Xr_{i,k}$ represents the amount of production steel of grade i in the time period k,while $Ur_{i,k}$ denotes the number of charges of grade i in the time period k. Eq. (28) states that the total amount of products per grade for each day should be equal to the pot size. To ensure there is always a numerical solution, we introduce slack variables $Sr1^{+}_{i,k}$ and $Sr1^{-}_{i,k}$, where $Sr1^{+}_{i,k}$ is equal to the amount of the leftover of grade i and time period k, while $Sr1^{-}_{i,k}$ becomes positive if the amount of steel products of grade i and time period k is greater than the total amount of all charges, which means that there are no physically feasible solutions.

$$Xr_{i,k} + Sr1^{+}_{i,k} - Sr1^{-}_{i,k} = PUr_{i,k} \quad \forall i,k \tag{28}$$

Eq. (29) constraints the number of pots for each day so that it is equal to the product of the number of pots per cast and the number of casts per day as shown below:

$$\sum_i Ur_{i,k} = UV \quad \forall k \tag{29}$$

with U and V being the number of casts per day and the number of charges per cast respectively. Then, instead of f_1^+ and f_1^-, we define new functions fr_1^+ and fr_1^- which represent the total leftover and the degree of violation of the pot capacity respectively as follows:

$$fr_1^+ = \sum_i \sum_k Sr1_{i,k}^+ \tag{30}$$

$$fr_1^- = \sum_i \sum_k Sr1_{i,k}^- \tag{31}$$

If a physical solution is infeasible related to the total production capacity, fr_1^- becomes positive, otherwise it is zero. If all charges are equal to the pot size, then fr_1^- is zero. Moreover, since Eqs. (16) and (25) include $X_{i,u,v,k}$, we can rewrite these equations as follows:

$$M_{i,k}^{in} - Xr_{i,k} = 0 \qquad \forall i,k \tag{32}$$

$$\sum_i \sum_{k',m:k'+m-1=k} Xr_{i,k} Load_{l,i,m} - Q_{l,k} = 0 \qquad \forall l,k \tag{33}$$

Eq. (33) imposes the constraint that makes the amount of production $Xr_{i,k'}$ over grade i with k' satisfying $k'+\Delta m = k$ equal to the production load $Q_{l,k}$ of process l on the time period k. With these relaxations, the relaxed mixed integer model is rewritten as follows:

$$\text{min} \quad W_1^+ fr_1^+ + W_1^- fr_1^- + W_3 f_3 + W_3^+ f_3^+ + W_3^- f_3^- + W_4^- f_4^- \tag{34}$$

s.t. (10)-(24), (17)-(33)

4.2 Meta-heuristic Approach to Charge Sequence Optimization

Solution Representation

One of the most widely used representations for sequence problem is job-to-position representation (Rhimi-Vahed and Mirzaei, 2007). The value of the first element represents a grade scheduled in the first charge. The second value shows a grade scheduled in the second charge and so on. Table 1 shows an example of solution representation. This example means that the first four charges are filled with products of grade A, and the following two charges are grade B. In the second cast, the first three charges are grade C, the next two charges are grade B and the last charge is grade D.

Table 1. Solution representation

Cast	**First cast**						**Second cast**					
Location	1	2	3	4	5	6	7	8	9	10	11	12
Grade	A	A	A	A	B	B	C	C	C	B	B	D

Simulated Annealing

In this work, we use simulated annealing (SA) in order to solve the scheduling problems. SA is a general meta-heuristic approach to combinatorial optimization (Aarts E & Korst J, 1989) and employs a neighbor search algorithm to move from one solution candidate to an improved solution in the neighborhood. Since the obtained solutions may be local optima, simulated annealing allows the objective value to worsen occasionally with a certain decreasing probability in order to avoid local minima as much as possible. That probability is specified by an acceptance probability function $P(f(\mathbf{x}), f(\mathbf{x}'), T)$ that depends on the two neighboring solutions **x** and **x**' and temperature T with f being the objective value. If $f(\mathbf{x}') \le f(\mathbf{x})$, we accept the solution **x**', otherwise we accept it with the probability of $\exp\left(\left(f(\mathbf{x}') - f(\mathbf{x})\right)/T\right)$. T is decreased at each step following some annealing schedules such as $T \leftarrow T \times T_{Factor}$ with T_{Factor} being a temperature factor. In this way, SA employs stochastic decision elements to search for global optimal solution as much as possible. Another possible approach is a Bayesian heuristic algorithm which also employs stochastic decision so that selecting neighboring solutions can achieve the better value of objective function. The neighboring solutions are selected at some probability which is computed by Bayesian method (Mockus & Reklaitis, 1999). The key

difference with respect to the Bayesian approach is to learn the parameters which control the probability. However, for parallel computing, the solution diversity is important factor to get better solutions and thus randomness is needed rather than learning parameters. Therefore, in this paper we employ SA method to solve the cast scheduling problems.

Since a reasonable solution is obtained from the relaxed MILP method, it is useful to use it as an initial state in the SA search. In order to obtain wide range of solutions, our search space is within the neighboring sequences that can be obtained by swapping two sets of charges not by swapping two single charges. Meanwhile, in order to avoid inefficient search as much as possible, we restrict the swapping as follows: (i) if two sets are swapped among different casts, their lengths are same, (ii) if swapped within the same cast, lengths of both sets are allowed to be different.

Initial solution

Since a reasonable solution is obtained from the relaxed MILP method, it is useful to use it as an initial state in the SA search. The solutions obtained from the relaxed MILP method denote the number of charges for each grade and each day and thus they do not represent the cast sequences. Therefore, we need to transform these solutions into the form of solutions that represent the sequence of charges. For this purpose, we optimize the sequence of charges for each day independently, and then all the obtained cast sequences are employed as an initial solution in the SA search as shown in Fig.5.

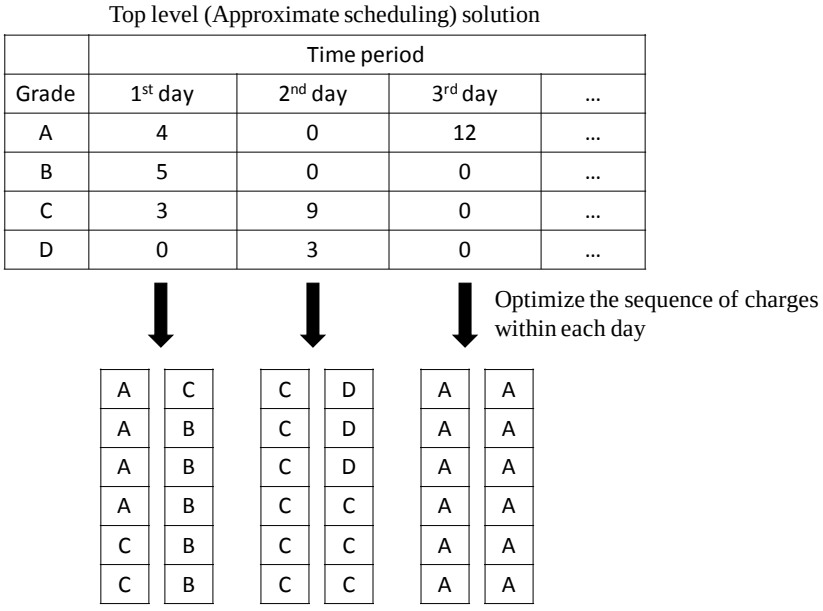

Fig. 5. Generation of an initial solution

Swapping Operator

In order to obtain wide range of solutions, our search space is within the neighboring sequences that can be obtained by swapping two sets of charges not by swapping two single charges. Meanwhile, in order to avoid inefficient search as much as possible, we restrict the swapping as: (i) if two sets are swapped among different casts, their lengths are same, (ii) if swapped within the same cast, lengths of both sets are allowed to be different. Therefore our searching spaces are restricted the two operations as described in Fig. 6.

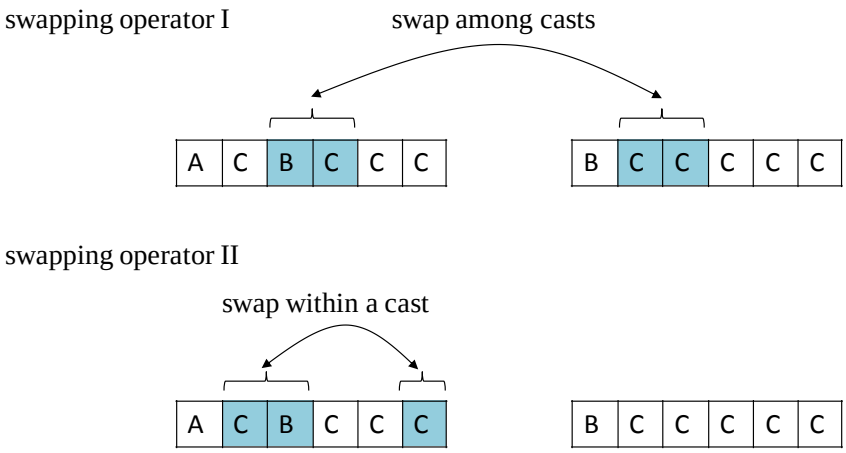

Fig. 6. Swapping operators

Evaluation of objective values in the SA search

It is computationally expensive to compute the value of the whole objective function every time after swapping sets of charges. Therefore, we re-compute only a part of objective values related to casts that are changed by a swapping operator, while the other objective values remain same. Table 2 shows the values of variables that should be re-computed after the swapping operator.

Table 2. Indices of the objective values that should be recomputed after a swapping operator

	Swapping within same cast; **time period = k1** **cast = u1.**	**Swapping among casts whose time periods are the same;** **time period = k1** **casts = u1,u2**	**Swapping among different casts whose time periods are different;** **time period = k1,k2** **casts = u1,u2**
$M_{i,k}^{in}$	$\forall i$, period k1	$\forall i$, cast u1 and u2	$\forall i$, cast u1 and u2
$Sr1_{i,k}^{+}$	$\forall i$, cast u1	$\forall i$, cast u1 and u2	$\forall i$, cast u1 and u2
$Sr1_{i,k}^{-}$	$\forall i$, cast u1	$\forall i$, cast u1 and u2	$\forall i$, cast u1 and u2
$MI_{i,k}^{open}$	$\forall i$, period k1	$\forall i$, period k1	$\forall i$, from period k1 to k2
$MI_{i,k}^{close}$	$\forall i$, period k1	$\forall i$, period k1	$\forall i$, from period k1 to k2
$Sr3_{i,k}^{+}$	$\forall i$, period k1	$\forall i$, period k1	$\forall i$, from period k1 to k2
$Sr3_{i,k}^{-}$	$\forall i$, period k1	$\forall i$, period k1	$\forall i$, from period k1 to k2

A Novel Parallel Simulated Annealing

In order to obtain better solutions, we consider parallel simulated annealing algorithms to get better solutions. Several researches have explored how to parallelize SA search (Ferreiro et al., 2013, Lee and Lee, 1996). The most straight-forward parallelization is to carry out SA on each worker independently and then to communicate their state at the end of process. This type of parallelization is called *asynchronous approach*. On the other hand, the *synchronous approaches* communicate states of all workers and states are exchanged among processors.

In this work, we improve the synchronous parallel SA in a manner that avoids local optima as much as possible by communicating states more effectively. The basic idea of this new version of parallel SA is that when the SA is converged in a worker, the new initial state is

generated from a combination of the best solution among all workers and the solution in the corresponding worker. The new initial states are computed by using the regeneration idea employed in the shuffled-frog leaping algorithm. The shuffled frog leaping algorithm is a memetic meta-heuristic that is based on the evolution of memes carried by individual and a global exchange of information among the population (Eusuff and et al., 2006). We use this evolutionary procedure in order to replace a local optimal solution with new one in each worker.

Let $\mathbf{P}_{\boldsymbol{B}}$ be the best solution and $\mathbf{P}_C(w)$ be the current solution of worker w, step size $\mathbf{S}$ is computed as follows:

$$\mathbf{S} = \min\left\{\text{int}\left[\text{rand}\left(\mathbf{P}_B - \mathbf{P}_C(w)\right)\right], \mathbf{S}_{\max}\right\} \text{ for a positive step} \quad (35)$$

$$\mathbf{S} = \min\left\{\text{int}\left[\text{rand}\left(\mathbf{P}_B - \mathbf{P}_C(w)\right)\right], -\mathbf{S}_{\max}\right\} \text{ for negative step} \quad (36)$$

where $\mathbf{S}_{max}$ denotes the maximum step which should be equal to the number of grades, rand is a random number in the range [0,1] and int returns the nearest integer. The new state is then computed by:

$$\mathbf{P}_C^{new}(w) = \mathbf{P}_C(w) + \mathbf{S} \quad (37)$$

Fig. 7 is a diagram of the proposed parallel simulated annealing method. First, we define K types of initial temperature and temperature factor and then assign them to each worker. K is the number of CPUs we can use. Then, the new solutions are computed by implementing the user specified number of steps (e.g. 2 or 3) of SA for each processor. Each processor performs the SA

process asynchronously. In order to increase the diversity along with the searching, all the solutions that are not improved take part in the evolution. If a new solution is better than the previous one, then accept the new solution. Otherwise, generate a solution using Eqs. (1)-(3). In this example, the solution $\mathbf{P}_C(2)$ is not improved in Worker 2 after implementing SA; it is replaced with the new state obtained by adding the step size $\mathbf{S}^{2,3}$ which is computed from Eq. (37) using the current state $\mathbf{P}_C(2)$ and the global state $\mathbf{P}_C(3)$. Similarly the unimproved solution of $\mathbf{P}_C(4)$ is replaced with the new state by adding the computed step size. Since the other solutions such as $\mathbf{P}_C(1)$ and $\mathbf{P}_C(3)$ are improved, they remain the same at this step. Using this procedure we update the solutions in each worker synchronously unless all the solutions are not improved or execution time becomes larger than the specified time.

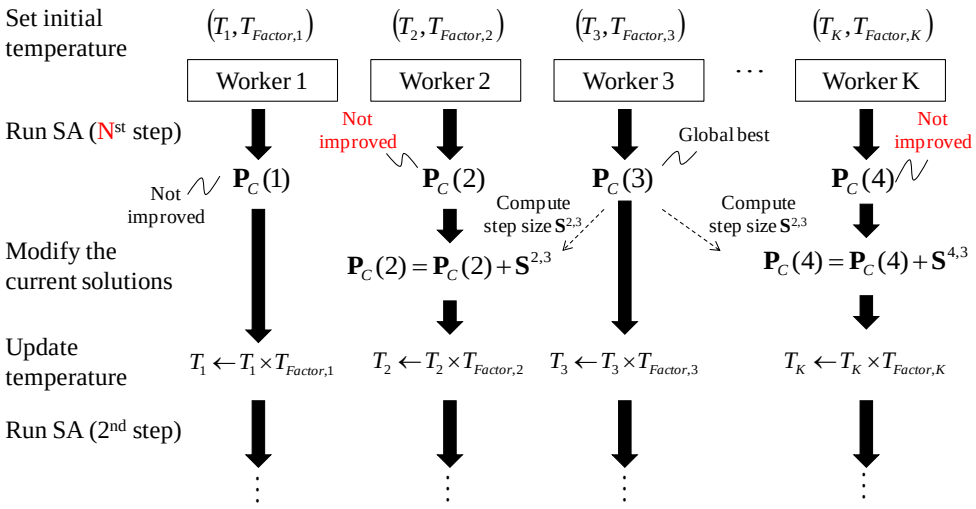

Fig. 7 Illustrative diagram of the proposed parallel simulated annealing

The new position itself does not represent a sequence because the elements are continuous values. Therefore, it is necessary to construct a sequence of grades from the new

position. We employ the rule (Jarboui et al., 2008, Liu et al., 2007) where the smallest position value of a charge is first picked and assigned grade A. Then, the second smallest position value of a charge is picked and assigned grade A if grade A is still available, otherwise assigned grade B, and so on. This procedureconverts the position information into a sequence of grades.

Table 3. Solution generation (A:3 charges, B:4 charges, C: 2 charges)

Location	1	2	3	4	5	6	7	8	9
New solution (continuous value)	2.2	2.2	1.6	2	2	2	1.4	1.8	1.8
New solution (Grade)	C	C	A	B	B	B	A	A	B

Convergence Criteria

We have employed two types of convergence criteria. First one is that if current best solution among all workers is not improved after implementing SA on each worker, the algorithm stops (Convergence Criterion I). Second one is that if all current solutions of each worker are not improved after implementing SA, algorithm stops (Convergence Criterion II). Compared to the convergence criterion I, the second criterion enables search for new solutions more deeply but computation times become larger.

The step by step procedure of the presented optimization approach is listed below.

1) Build the Bayesian network model that can predict the probability distributions of the production loads and process times from historical process data.
2) Estimate the probability distribution of production loads and process time for each order (plate) by means of Bayesian inference method based on the constructed Bayesian network model.

3) Compute the total demand of each grade on the ideal production starting day and the expectation of production load of each grade at each process unit and each elapsed day.
4) Determine grades which should be produced in every time period to satisfy the customer due dates and production capacities by solving the relaxed mixed integer problem at the planning stage through the commercial solver such as CPLEX.
5) Set the initial sequence using the solution generated in step 4).
6) Define *K* types of initial temperature and temperature factor and assign them to each worker with K being the number of CPU we can use.
7) Find the sequence of charges from the current sequence by implementing specific number of steps of SA for each processor which performs a SA process asynchronously.
8) If a new solution (sequence) is better than the previous one, then accept new solution. Otherwise, generate a solution based on the idea of frog flip leaping algorithm.
9) Discretize the new solution.
10) If all workers do not improve their solutions, then the algorithm is terminated. Otherwise, return to step 7).

5. Case Studies

In this section, the results obtained by the proposed parallel SA with frog leaping are presented. All the case studies have been computed on a windows 2007 Professional (Intel® Core(TM) i7-4930 CPU @ 3.40GHz and 16.0 Gb RAM). The full Space and the top level production

planning models were solved using Gurobi 5.6.3.

5.1 Small test problems

Three small test problems are used to see how far from the true optimum are the solutions obtained from our algorithm for the smaller size problems. The optimum is computed by solving the full space MILP model. Table 4 shows the size of full-space MILP model.

Table 4. Size of three small test problems

Example	**Horizon [days]**	**Total Term [days]**	**# casts**	**# grades**	**Full space MILP model**	
					# Conts	**# Bins**
1	7	10	7	3	526	126
2	5	7	5	7	1,537	210
3	6	8	6	7	1,857	252

Horizon=length of days for which the production scheduling are made, total term=length of days for which the production scheduling deals with the orders, #casts = number of casts, #grades = number of casts, #Conts = number of continuous variables, and #Bins = number of binary variables

Table 5 compares the results from the full space model MILP, the single SA and the parallel SA methods and the proposed parallel SA with shuffled frog leaping algorithm. In these examples, the optimality gaps obtained from full space MILP models are 0% and the objective function values of both full space MILP and the proposed parallel SA with shuffled frog leaping algorithms are same. In example 1, the execution time required by the full space MILP algorithm is smaller than the proposed method. However, sa the problem size increases, the proposed algorithm becomes much faster than the full space MILP, as shown by examples 2 and 3.

Table 5. Results comparison between solution algorithms for small test problems

Test sets	Algorithm	CPU time [s]	Obj. Func. Values	Lower Bound	Gap(%)
1	Full space MILP	4	700.00	700.00	0.0
	Single SA*1	19	700.00	-	-
	Parallel SA	13	700.00	-	-
	Single SA*1 (two level algorithm)	22	700.00	-	-
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	97	700.00	-	-
2	Full space MILP	466	300.00	300.00	0.0
	Single SA*1	16	500.00	-	-
	Parallel SA	14	500.00	-	-
	Single SA*1 (two level algorithm)	11	400.00	-	-
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	89	300.00	-	-
3	Full space MILP	99,172	348.03	348.03	0.0
	Single SA*1	29	400.00	-	-
	Parallel SA	20	400.00	-	-
	Single SA*1 (two level algorithm)	22	400.00	-	-
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	227	348.03	-	-

*1 Initial temperature T=0.8 and temperature factor T_{Factor} is set to be the best value among [0.5, 0.6, 0.7, 0.8, 0.9]

5.2 Real-world steel-plate production scheduling problems

The real-world steel-plate production data are utilized to examine the effectiveness of the proposed optimization algorithm. The two test sets we consider are based on one week plan of a steel-plate production and the number of grades being 23 and 36 respectively. The production capacities and the size of full space MILP models are shown in Table 6 and 7 respectively.

Table 6: Process capacity

Mill	**Processes**	**Production Capacity**
Steel making mills	# casts	3[casts per day]
	# charges	6[charges per cast]
	Size of a Charge	200[ton/charge]
Steel plate making mills	Process A	350[ton/day]
	Process B	700[ton/day]
	Process C	1200[ton/day]
	Process D	500[ton/day]
	Process E	400[ton/day]
	Process F	100[ton/day]
	Process G	1500[ton/day]
	Process H	150[ton/day]
	Process I	100[ton/day]

#casts = number of casts and #charges = number of charges

Table 7. Size of two industrial test problems

Test sets	**Horizon** [days]	**Total term**	**# casts**	**# grades**	**Full space MILP model**	
					# Conts	**# Bins**
1	7	13	3	23	59,270	2,898
2	7	13	3	36	141,866	4,536

Two-level algorithm both with single SA and the proposed parallel SA are used to compute solutions. As a comparison, the full space model MILP, the single SA, and the parallel SA methods are also carried out to compute the solutions. As for parallel computing in parallel SA and MILP, 12 parallel workers are used.

Test Set #1

Test Set #1 problem computes the first level SCC schedule for 23 kinds of steel grades and the scheduling horizon is one week. Table 8 compares the results from full space model MILP, single SA, parallel SA, two-level algorithm with single SA, and two-level algorithm with the new parallel SA. It can be seen that the full space model algorithm computes the worst

objective values even though the computation time is set to be 24 hours (86400[s]). That is because the problem is NP-hard; clearly, the full space model MILP algorithm does not work for industrial-scale problems.

On the other hand, meta-heuristic approaches give us much better solutions in terms of both values of objective function and execution times as shown in Table 8. For instance, the single SA method shows lower objective values and its computational time is much less than the solution of the full-space model MILP. Further, objective values of the proposed two-level algorithm with single SA are even better than those of the simple SA approach, since the relaxed model, which takes into account all the objective functions except for the sequence penalty, gives good initial solutions for the SA search. This comparison shows that the proposed two-level algorithm with single SA has ability to find better solutions than both the full space model MILP approach and the single SA method.

As for the parallel computing methods, we have investigated the effectiveness of the previously published parallel SA (Ferreiro et al., 2013), where all workers start from random initial solutions so that each worker runs independently SA until reaching the next level of temperature. The objective value of the parallel SA is better than all the above methods due to multi-initial solutions. The proposed two-level algorithm with a new parallel SA at the lower level has the elapsed time of 393 seconds when the convergence criterion I is satisfied; the objective value is 3813.97 (Table 8), which means that both objective value and execution time are less than all the above methods. In addition, when the convergence criterion II is used, the elapsed time is 1,268.9 seconds elapsed and the objective value becomes 3,715.21. This is the lowest among all algorithms evaluated in this work. The proposed two-level algorithm with the

new parallel SA with leaping has the best performance in terms of both the objective value and the execution time.

The scheduling results obtained from the proposed parallel SA with frog leaping algorithm are shown in Figs. 8 to 10. Fig. 8 shows the cast scheduling where horizontal axis represents the time period and the vertical axis denotes the grade index. Each number represents a grade (e.g. 1 means grade A, 2 means grade B and so on). Fig. 7 shows the trend plots of inventories for each grade. In Fig. 8 the trend plots of production loads of finishing lines and their production capacities are drawn by blue solid lines and red dashed lines respectively. It should be pointed out that all production loads are less than their production capacity for the entire planning period and almost all of inventory levels are greater than zero.

Table 8. Comparison of solution algorithms for real steel-plate production scheduling problems

Test sets	**Algorithm**	**Execution time[s]**	**Obj. Func. Values**	**Lower Bound**	**Gap(%)**
1	Full space MILP	86,400	5,979.24	0	>100
	Single SA*1	429	5152.06	-	-
	Parallel SA	399	4508.39	-	-
	Single SA*1 (two level algorithm)	466	4615.08	-	-
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	393	3,831.97	-	-
	Parallel SA with leaping algorithm Convergence Criterion II (two level algorithm)	1,905	3,715.21	-	--
2	Full space MILP	86,400	21,201.46	0	>100
	Single SA*1	368	18,619.67	-	-
	Parallel SA	628	18,376.50	-	-
	Single SA*1 (two level algorithm)	313	18,179.84	-	-
	Parallel SA with leaping algorithm Convergence Criterion I (two level algorithm)	508	17,540.18	-	-
	Parallel SA with leaping algorithm Convergence Criterion II (two level algorithm)	2,951	17,463.44	-	-

*1 Initial temperature T=0.8 and temperature factor T_{Factor} is set to be the best value among [0.5, 0.6, 0.7, 0.8, 0.9]

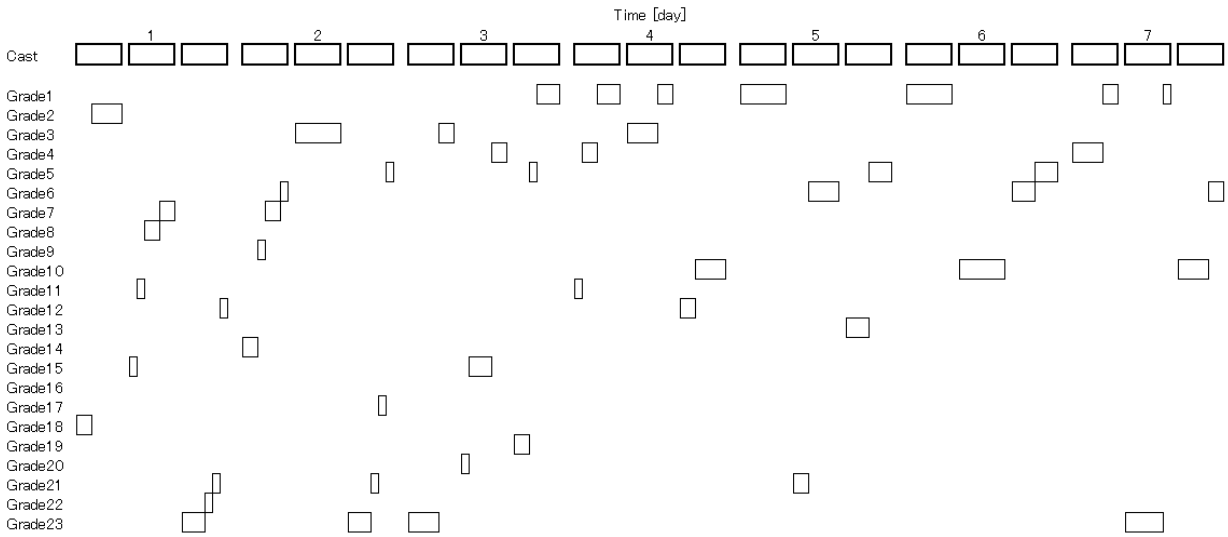

Fig. 8 Cast schedule computed by two-level algorithm with the proposed parallel SA for test set #1

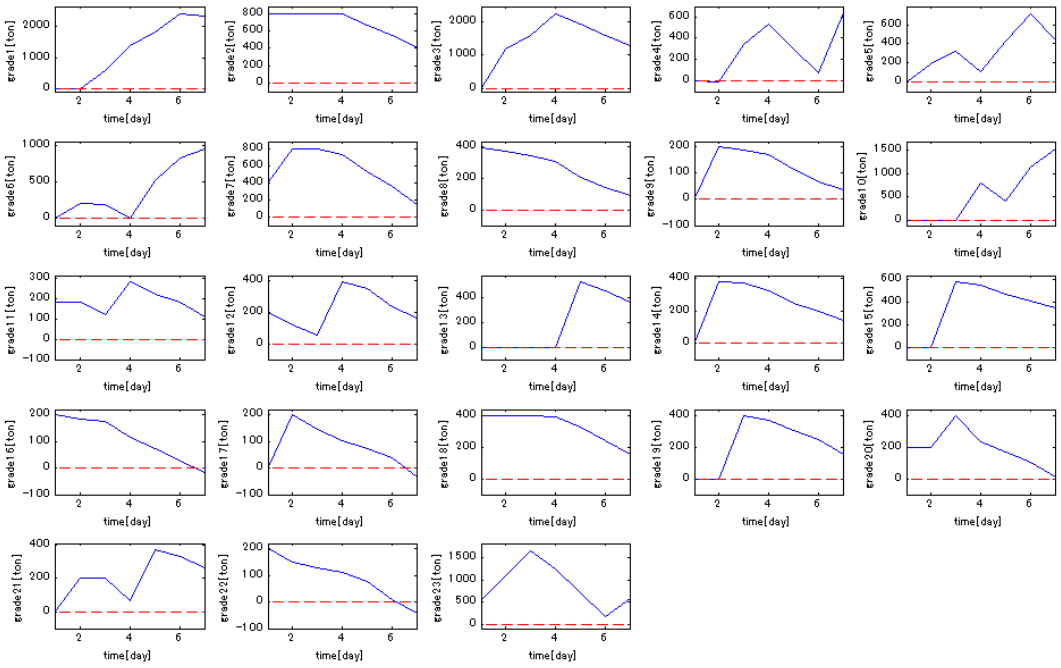

Fig. 9 Trend plots of inventory computed by two-level algorithm with the proposed parallel SA for test set #1

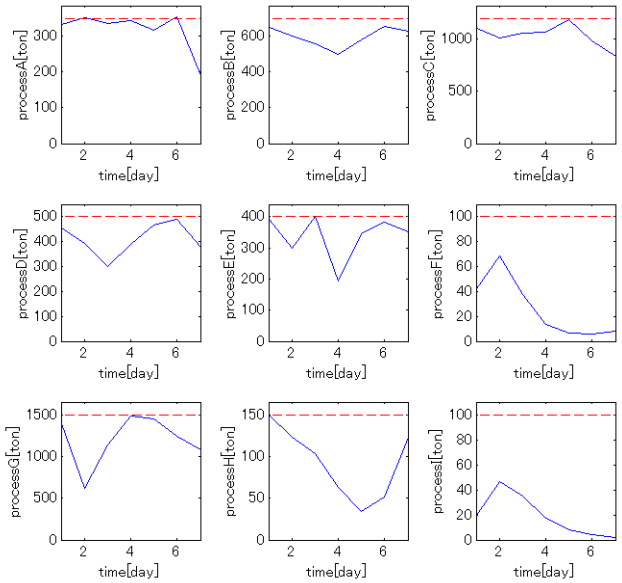

Fig. 10 Trend plots of production loads by two-level algorithm with the proposed parallel SA for test set #1

Test Set #2

Test Set #2 is different from test set #1 only in the number of grades which need to be scheduled, as shown in Table 7. The results from the full space model MILP, the single SA, the parallel SA, the two-level algorithm with single SA, and the two-level algorithm with the new parallel SA with frog leaping are shown in Table 8. Similarly to test set #1, the objective value of the full space model MILP is the worst among all the above methods even though its maximum execution time is set to be 24 hours (86400[s]).

As for meta-heuristic approaches, the objective value and the execution time of single SA are much lower than those of the full space model MILP approach. In addition, the proposed two-level algorithm with single SA performs better than those of the simple SA approach in terms of both objective value and its execution time than the single SA due to the same reason mentioned in the test set #1. This confirms the effectiveness of the two-level algorithm due to the relaxed model MILP creating good initial solutions for SA search.

The parallelization of SA given random initial solution set for each worker gives us slightly lower objective values than the single SA while its computational time is about twice as long as single SA. When applied within the two-level, parallel methods yield results shown in Table 8. When the convergence criterion I is satisfied at the time of 508 seconds, the solution obtained has the objective value of 17,540.18, which is the lower than the above methods. Furthermore, when the convergence criterion II applied, the objective function value becomes 17,463.44, which is the smallest in our experiments for test set #2. It should be noted that the proposed two-level algorithm with the new parallel SA with leaping provides the best scheduling performance in terms of both the objective value and the execution time. The scheduling results implemented by the two-level algorithm via new parallel SA are shown in Figs. 11 to 13. Similarly to the test set #1, the inventories are greater than zero and the production loads are less than their capacities for the whole planning horizon.

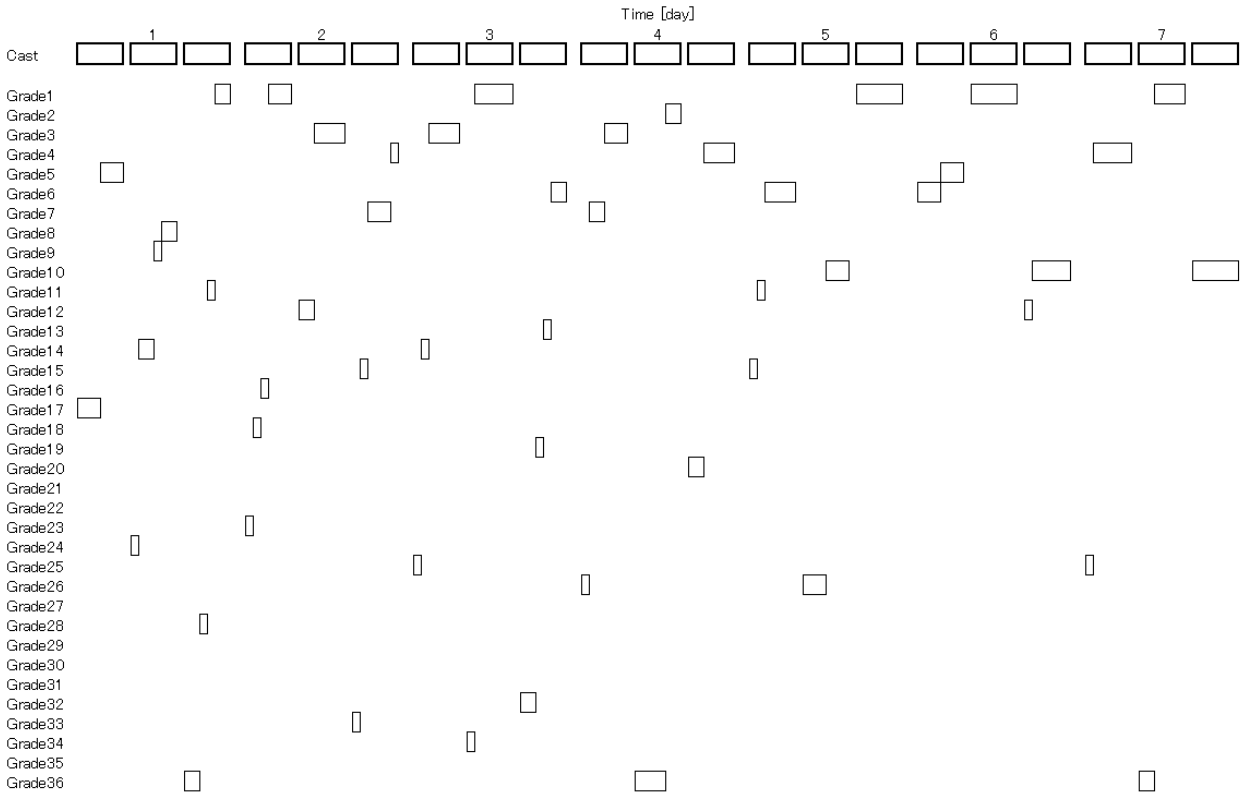

Fig. 11 Cast schedule computed by two-level algorithm with the proposed parallel SA for test set #2

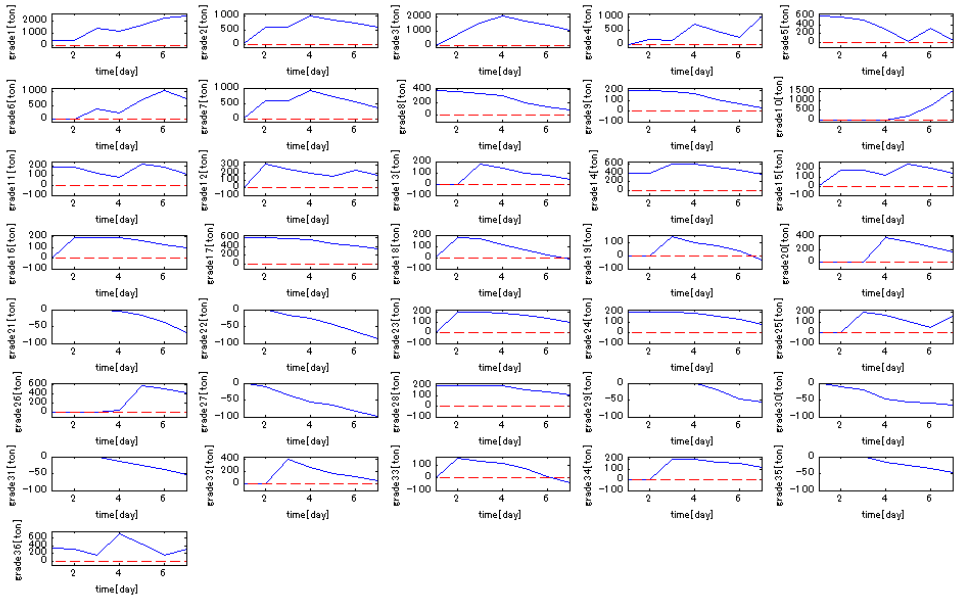

Fig. 12 Trend plots of inventory computed by two-level algorithm with new parallel SA for test set #2

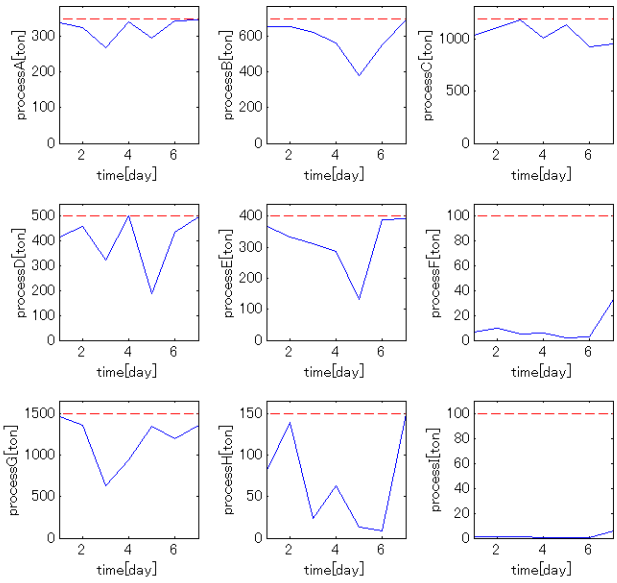

Fig. 13 Trend plots of production loads computed by two-level algorithm with new parallel SA for test set #2

6. Conclusions

This work proposes a new optimization algorithm which can assign and sequence a large number of different grades of steel plates into a continuous casting machine for steel plate production. In order to consider the production loads of downstream processes and production times, Bayesian inference method is utilized to predict their probability distributions from customer demands and operating conditions. Bayesian inference is derived from the Bayesian network model developed from the historical process data as described in our preceding work. Continuous casting scheduling is a difficult optimization problem due to a large number of binary variables which are needed to represent exactly process characteristics in the optimization model. In order to reduce compute high quality schedules, we propose a new two-level algorithm.

At the top level, we solve a multi-period production planning (it can be considered as approximate scheduling) which is a mixed-integer linear model that does not take into account sequence penalties. Solution of this problem is guaranteed to be feasible at the period boundaries and it is easy to compute via present days MILP solvers. The lower level is tasked with computing the sequencing of charges such that there are a minimum number of switches. This is accomplished by permuting the sequences within the constraints of the top level solution.

We have tested several different variations of the two level algorithm. They differ with respect to implementation of the sequencing at the lower level. One version employs simulated annealing, while the other employ modified parallel simulated annealing combined with leaping ideas from shuffled frog algorithm. Simulated annealing has been selected to solve this sequence optimization problem because the SA method has strong capability to avoid local optimal solutions as much as possible by allowing the objective values to worsen occasionally.

Proposed new parallelization approach to SA search aims to further improve the ability of SA to avoid local optimal solutions. The main concept of the proposed parallel SA is that unimproved solutions on each SA step are replaced with new solutions which are computed from the current solution in the corresponding worker and a global best solution among all workers. During the computation of the new solutions, we incorporate into the parallel SA the ideas that have been proposed in the shuffled-frog leaping algorithm.

For small scale test problems, which can be solved to 0.0% optimality gap via full space MILP model, the proposed algorithm computes the global optimum.

Real world steel production data have been used to examine the effectiveness of the presented optimization approach on large problems which arise in industrial practice. The computational results demonstrate that the proposed two-level algorithm generates better solutions and within shorter execution times than both the single SA and the existing parallel SA. Even though the proposed meta-heuristic optimization method is not guaranteed to yield global optimal solutions, it can generate the much better solutions within practical execution times than the rigorous full-space MILP method over much longer execution times.

Nomenclature

Indices

I	Index of grades
U	Index of casts
V	Index of charges
K	Index of time period (day)
L	Index of finishing and inspection processes

Parameters

W	Cost coefficient in the objective function
$D_{i,k}$	Demand of grade I on time period k
$C_{i,k}$	Contamination cost when grade i is assigned after grade i’
P	Maximum amount of charges
MI_{max}	Maximum inventory
MI_{min}	Minimum inventory
$Load_{l,i,m}$	Production load of process unit l, grade i, and elapsed day m
$Load_{l}^{\max}$	Production capacity of process unit l

Binary variables

$Y_{i,u,v,k}$	Binary variables to determine if grade i is assigned in charge v of cast u on time period k
$Z_{i,i',u,v,k}$	Binary sequence variables between grade i is assigned in charge v of cast u on time period k

Integer variables

$Ur_{i,k}$	Integer variables for the number of charges of grade i on time period k

Continuous variables

$X_{i,u,v,k}$	Continuous variables for amount of grade i, cast u, charge v on time period k
$M_{i,k}^{in}$	Continuous variables for amount of grade i on time period k
$M_{i,k}^{out}$	Continuous variables for demand of grade i on time period k
$M_{i,k}^{open}$	Continuous variables for opening volume of grade i on time period k
$M_{i,k}^{close}$	Continuous variables for closing volume of grade i on time period k
$Q_{l,k}$	Continuous variables for production loads of process unit l on time period
$S1_{i,u,v,k}^{+}$	Positive slack variables for production capacity at the continuous casting stage
$S1_{i,u,v,k}^{-}$	Negative slack variables for production capacity at the continuous casting stage
$S3_{i,k}^{+}$	Positive slack variables for maximum inventory capacity
$S3_{i,k}^{-}$	Negative slack variables for minimum inventory capacity

$S4^{-}_{I,k}$	Negative slack variables for production capacity at the finishing and inspection processes
$Xr_{i,k}$	Continuous variables for amount of grade *i* on time period *k*
$Sr1^{+}_{i,k}$	Positive slack variables for production capacity at the continuous casting stage
$Sr1^{-}_{i,k}$	Negative slack variables for production capacity at the continuous casting stage
$S^{+}_{pr,L3}(j,n), S^{-}_{pr,L3}(j,n)$	Positive and negative slack variables for the product inventories

References

Aarts E, Korst J. Simulated Annealing and Boltzmann Machines: A Stochastic Approach to

Combinatorial Optimization and Neural Computing. New York: John Wiley and Sons, Inc. 1989.

Atighehchian A, Bijari M, Tarkesh H. A novelhybridalgorithmforschedulingsteel-makingcontinuouscastingproduction. 2009; 36: 2450-2461.

Dash S, Kalagnanam J, Reddy C, Song S. Production design for plate products in the steelindustry. IBM journal of research and development. 2007; 51: 345-362.

Eusuff M, Lansey K, Pasha F. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Eng Optim. 2006; 38: 129-154.

Ferreiro A, Garcia J, Lopez-Salas JG, Vazquez C. An efficient implementation of parallel simulated annealing algorithm in GPUs. J Glob Optim. 2013; 57: 863-890.

Harjunkoski I, Grossmann I.E. A decomposition approach for the scheduling of a steel plant production. Comput Chem Eng. 2001; 25: 1647-1660.

Glover F. Future paths for integer programming and links to artificial intelligence. Comput Oper Res. 1986; 13: 533-549.

Jarboui B, Damak N, Siarry P, Rebai A. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Applied Math and Comput. 2008; 195: 299-308.

Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of IEEE Conference on Neural Networks. 1995; 1942-1948.

Lawrence D. vol. 115. New York: Van Nostrand Reinhold. 1991.

Lee S, Lee K. Synchronous and asynchronous parallel simulated annealing with multiple Markov

Chains. IEEE Trans Parallel Distrib Syst. 1996; 7: 993-1008.

L. Mockus, G. V. Reklaitis. Continuous time scheduling representation approach to batch and continuous process scheduling. 2. Computational Issues. Ind. Eng. Chem. Res. 1999; 38: 204-210

Liu B, Wang L, Jin YH. An effective PSO-based memetic algorithm for flow shop scheduling. IEEE Trans Syst, Man, Cybern, Syst. 2007; 37: 18-27.

Lourenco H, Martin O, Stutzle T. Iterated Local Search. Kluwer. 2002.

Marco D, Birattari M. Ant colony optimization. SpringerVerlag. 2010.

Moon S, Hrymak A. Scheduling of the batch annealing process - deterministic case. Comput Chem Eng. 1999; 23: 1193-1208.

Mori J, Mahalec V. Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables. Comput Chem Eng 2015; 79: 113-134

Rahimi-Vahed A, Mirzaei A. A hybrid multi-objective shued frog-leaping algorithm for a mixedmodel assembly line sequencing problem. Comput Ind Eng. 2007; 53: 642-666.

Santos C, Spim J, Garci A. Mathematical modeling and optimization strategies (genetic algorithm and knowledge base) applied to the continuous casting of steel. Eng Appl Artif Intel. 2003; 16: 11-527.

Schaus P, Hentenryck P, Monette JN, Coffrin C, Michel L, Deville Y. Solving Steel Mill Slab Problems with Constraint-Based Techniques: CP, LNS, and CBLS. Constraints. 2010; 16: 125-147.

Tang L, Luh P.B, Liu J, Gang L. Steel-making process scheduling using Lagrangean relaxation. INT. J. PROD. RES., 2002; 40: 55-70.

Tang L, Liu J, Rong A, Yang Z. A mathematical programming model for scheduling steelmaking continuous casting production. Eur J Oper Re. 2000; 120: 423-435.